

```

BBBBBBBBBBBBBB      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
BBBBBBBBBBBBBB      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
BBBBBBBBBBBBBB      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAAAAAAAAAAAAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAAAAAAAAAAAAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAAAAAAAAAAAAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSSSSSSSSSSS      RRR      RRR      TTT      TTT      LLLLLLLLLLLLLLLL
BBBBBBBBBBBBBB      AAA      AAA      SSSSSSSSSSSS      RRR      RRR      TTT      TTT      LLLLLLLLLLLLLLLL
BBBBBBBBBBBBBB      AAA      AAA      SSSSSSSSSSSS      RRR      RRR      TTT      TTT      LLLLLLLLLLLLLLLL

```

```
BBBBBBBB  AAAAAA  SSSSSSSS  CCCCCCCC  VV  VV  TTTTTTTTTT  TTTTTTTTTT  PPPPPPPP
BBBBBBBB  AAAAAA  SSSSSSSS  CCCCCCCC  VV  VV  TTTTTTTTTT  TTTTTTTTTT  PPPPPPPP
BB  BB  AA  AA  SS  SS  CC  CC  VV  VV  TT  TT  PP  PP
BB  BB  AA  AA  SS  SS  CC  CC  VV  VV  TT  TT  PP  PP
BB  BB  AA  AA  SS  SS  CC  CC  VV  VV  TT  TT  PP  PP
BB  BB  AA  AA  SS  SS  CC  CC  VV  VV  TT  TT  PP  PP
BBBBBBBB  AA  AA  SSSSSS  CC  CC  VV  VV  TT  TT  PPPPPPPP
BBBBBBBB  AA  AA  SSSSSS  CC  CC  VV  VV  TT  TT  PPPPPPPP
BB  BB  AAAAAAAAAA  SS  CC  VV  VV  TT  TT  PP
BB  BB  AAAAAAAAAA  SS  CC  VV  VV  TT  TT  PP
BB  BB  AA  AA  SS  CC  VV  VV  TT  TT  PP
BB  BB  AA  AA  SS  CC  VV  VV  TT  TT  PP
BBBBBBBB  AA  AA  SSSSSSSS  CCCCCCCC  VV  VV  TT  TT  PP
BBBBBBBB  AA  AA  SSSSSSSS  CCCCCCCC  VV  VV  TT  TT  PP
```

```
LL  I I I I I  SSSSSSSS
LL  I I I I I  SSSSSSSS
LL  I I  SS
LL  I I  SS
LL  I I  SS
LL  I I  SS
LL  I I  SSSSSS
LL  I I  SSSSSS
LL  I I  SS
LL  I I  SS
LL  I I  SS
LLLLLLLLLL  I I I I I  SSSSSSSS
LLLLLLLLLL  I I I I I  SSSSSSSS
```

```
0001 0 ZTITLE 'BASSCVT_T_P - convert numeric text to packed decimal'
0002 0 MODULE BASSCVT_T_P (
0003 0 IDENT = '1-010'
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *
0028 1 *****
0029 1
0030 1 ++
0031 1 FACILITY: BASIC Language Support
0032 1
0033 1 ABSTRACT:
0034 1
0035 1 This routine converts numeric text to packed decimal.
0036 1
0037 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0038 1
0039 1 AUTHOR: Pamela L. Levesque, CREATION DATE: 29-Dec-1981
0040 1
0041 1 MODIFIED BY:
0042 1
0043 1 1-001 - Original. PLL 29-Dec-1981
0044 1 1-002 - Declare BASS Psects. PLL 11-Feb-1982
0045 1 1-003 - Incorporate changes based on code review. PLL 11-Feb-1982
0046 1 1-004 - Fix bug in calculation of integer and fraction digits. PLL 12-Feb-1982
0047 1 1-005 - If the input string is all blanks, return 1 integer zero and
0048 1 the appropriate number of fractional zeroes.
0049 1 Also incorporated DGP's suggestions. PLL 17-Feb-1982
0050 1 1-006 - Fix a bug in the all blanks case. PLL 29-Mar-1982
0051 1 1-007 - Look at a flag in the frame to determine rounding or truncating.
0052 1 PLL 10-Jun-1982
0053 1 1-008 - Remove edit 007 - it's the caller's responsibility to check the
0054 1 flag in the frame and pass it as an argument if necessary. PLL 30-Jun-1982
0055 1 1-009 - when skip_tabs flag is set, the spanc mask should be set to
0056 1 skip_blanks OR skip_tabs, not skip_blanks AND skip_tabs. MDL 30-Jun-1983
0057 1
```



BASSCVT\_T\_P  
1-010

BASSCVT\_T\_P - convert numeric text to packed de

F 10  
16-Sep-1984 00:17:11  
14-Sep-1984 11:54:49

VAX-11 Bliss-32 V4.0-742  
[BASRTL.SRC]BASCVTTP.B32;1

Page 2  
(1)

:	58	0058	1	:	1-010 - check return status from LIB\$\$ADDP_R7. MDL 13-Jan-1984
:	59	0059	1	:	--
:	60	0060	1	:	

```

62 0061 1 %SBTTL 'Declarations'
63 0062 1
64 0063 1 SWITCHES:
65 0064 1
66 0065 1
67 0066 1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
68 0067 1
69 0068 1
70 0069 1 LINKAGES:
71 0070 1
72 0071 1
73 0072 1 LINKAGE
74 0073 1 JSB_R7 = JSB (REGISTER = 4, REGISTER = 5, REGISTER = 6, REGISTER = 7) :
75 0074 1 PRESERVE (8, 9, 10, 11);
76 0075 1
77 0076 1 TABLE OF CONTENTS:
78 0077 1
79 0078 1
80 0079 1 FORWARD ROUTINE
81 0080 1 BASSCVT_T_P; ! convert from text to packed
82 0081 1
83 0082 1
84 0083 1 INCLUDE FILES:
85 0084 1
86 0085 1
87 0086 1 LIBRARY 'RTLSTARLE'; ! System symbols
88 0087 1
89 0088 1 REQUIRE 'RTLIN:RTLPSECT'; ! Define PSECT declarations macros
90 0183 1
91 0184 1
92 0185 1 MACROS:
93 0186 1
94 0187 1
95 0188 1 MACRO
96 0189 1 ADDP = LIB$$ADDP_R7 %;
97 0190 1
98 0191 1
99 0192 1 EQUATED SYMBOLS:
100 0193 1
101 0194 1 NONE
102 0195 1
103 0196 1 FIELDS:
104 0197 1
105 0198 1 NONE
106 0199 1
107 0200 1 PSECTS:
108 0201 1
109 0202 1
110 0203 1 DECLARE_PSECTS (BAS);
111 0204 1
112 0205 1
113 0206 1 OWN STORAGE:
114 0207 1
115 0208 1 NONE
116 0209 1
117 0210 1 EXTERNAL REFERENCES:
118 0211 1
```

BAS\$CVT\_T\_P  
1-010

BAS\$CVT\_T\_P - convert numeric text to packed de  
Declarations

H 10  
16-Sep-1984 00:17:11  
14-Sep-1984 11:54:49

VAX-11 Bliss-32 V4.0-742  
[BASRTL.SRC]BAS\$CVTTP.B32;1

Page 4  
(2)

:	119	0212	1	
:	120	0213	1	EXTERNAL ROUTINE
:	121	0214	1	LIB\$\$ADDP_R7 : JSB_R7;
:	122	0215	1	
:	123	0216	1	

```
125 0217 1 XSBTTL 'BASSCVT_T_P - convert numeric text to packed decimal'
126 0218 1 GLOBAL ROUTINE BASSCVT_T_P ( convert numeric text to packed decimal
127 0219 1 TEXT_DSC, text descriptor
128 0220 1 PACKED_DSC, packed descriptor
129 0221 1 FLAGS user flags
130 0222 1 ) =
131 0223 1
132 0224 1 ++
133 0225 1 FUNCTIONAL DESCRIPTION:
134 0226 1
135 0227 1 Converts a text string representing a numeric value to a packed
136 0228 1 decimal value. Description of the text is as follows:
137 0229 1 <0 or more spaces, nulls, or tabs>
138 0230 1 <"+ or "- or nothing>
139 0231 1 <0 or more digits>
140 0232 1 <". or nothing>
141 0233 1 <0 or more digits>
142 0234 1 <0 or more spaces, nulls, or tabs>
143 0235 1 <end of string>
144 0236 1
145 0237 1 Setting the strip spaces bit in FLAGS causes embedded spaces to be ignored.
146 0238 1 Otherwise blanks are an error.
147 0239 1
148 0240 1 Setting the skip tabs bit in FLAGS causes embedded tab characters to be ignored.
149 0241 1 Otherwise they are an error.
150 0242 1
151 0243 1 The value will be rounded, if necessary, to fit into the number of
152 0244 1 digits specified by the packed decimal descriptor. Truncation may be
153 0245 1 requested by setting the don't round bit in FLAGS.
154 0246 1
155 0247 1 Note that this routine expects the length and scale fields in the
156 0248 1 packed decimal descriptor to be set by the caller.
157 0249 1
158 0250 1
159 0251 1 CALLING SEQUENCE:
160 0252 1
161 0253 1 ret_status.wlc.v = BASSCVT_T_P (TEXT_DSC.rt.dx1, PACKED_DSC.wp.dsd, FLAGS.rl.v)
162 0254 1
163 0255 1 FORMAL PARAMETERS:
164 0256 1
165 0257 1 TEXT_DSC address of the input text descriptor
166 0258 1 PACKED_DSC address of the output decimal descriptor
167 0259 1 [FLAGS] value of caller's flags
168 0260 1 bit 0 set indicates skip blanks
169 0261 1 bit 3 set indicates don't round value
170 0262 1 bit 4 set indicates skip tabs
171 0263 1
172 0264 1 IMPLICIT INPUTS:
173 0265 1
174 0266 1 NONE
175 0267 1
176 0268 1 IMPLICIT OUTPUTS:
177 0269 1
178 0270 1 NONE
179 0271 1
180 0272 1 ROUTINE VALUE:
181 0273 1
```



```
182 0274 1 | success text successfully converted to decimal
183 0275 1 | failure numeric text could not be converted to the
184 0276 1 | specified decimal descriptor
185 0277 1 |
186 0278 1 | SIDE EFFECTS:
187 0279 1 |
188 0280 1 | NONE
189 0281 1 |
190 0282 1 | --
191 0283 1 |
192 0284 2 | BEGIN
193 0285 2 |
194 0286 2 | MAP
195 0287 2 | TEXT_DSC : REF BLOCK [8,BYTE],
196 0288 2 | PACKED_DSC : REF BLOCK [12,BYTE];
197 0289 2 |
198 0290 2 | BUILTIN
199 0291 2 | ACTUALCOUNT,
200 0292 2 | CVTSP,
201 0293 2 | SPANC;
202 0294 2 |
203 0295 2 | LITERAL
204 0296 2 | K_FLAGS_ARG = 3,
205 0297 2 | K_PACKED_ONE = %X'0000001C',
206 0298 2 | K_PACKED_NEG_ONE = %X'0000001D',
207 0299 2 | K_MAX_DIGITS = 31,
208 0300 2 | CHAR_TAB = %X'09' : UNSIGNED (8),
209 0301 2 | CHAR_SPACE = %X'20' : UNSIGNED (8),
210 0302 2 | CHAR_NULL = %X'00' : UNSIGNED (8),
211 0303 2 | CHAR_ZERO = %X'30' : UNSIGNED (8),
212 0304 2 | CHAR_ONE = %X'31' : UNSIGNED (8),
213 0305 2 | CHAR_TWO = %X'32' : UNSIGNED (8),
214 0306 2 | CHAR_THREE = %X'33' : UNSIGNED (8),
215 0307 2 | CHAR_FOUR = %X'34' : UNSIGNED (8),
216 0308 2 | CHAR_FIVE = %X'35' : UNSIGNED (8),
217 0309 2 | CHAR_SIX = %X'36' : UNSIGNED (8),
218 0310 2 | CHAR_SEVEN = %X'37' : UNSIGNED (8),
219 0311 2 | CHAR_EIGHT = %X'38' : UNSIGNED (8),
220 0312 2 | CHAR_NINE = %X'39' : UNSIGNED (8),
221 0313 2 | MASK_NUMERIC = 1 : UNSIGNED (8),
222 0314 2 | MASK_BLANKS = 2 : UNSIGNED (8),
223 0315 2 | MASK_TAB = 4 : UNSIGNED (8),
224 0316 2 | V_SKIP_TABS = 1^4,
225 0317 2 | V_SKIP_BLANKS = 1^0,
226 0318 2 | V_DONT_ROUND = 1^3;
227 0319 2 |
228 0320 2 | BIND
229 0321 2 |
230 0322 2 | SPANC_TABLE = UPLIT BYTE
231 0323 2 | (
232 0324 2 | REP CHAR_NULL OF (0),
233 0325 2 | (MASK BLANKS),
234 0326 2 | REP CHAR_TAB - CHAR_NULL - 1 OF (0),
235 0327 2 | (MASK TAB),
236 0328 2 | REP CHAR_SPACE - CHAR_TAB - 1 OF (0),
237 0329 2 | (MASK BLANKS),
238 0330 2 | REP CHAR_ZERO - CHAR_SPACE - 1 OF (0),
```



```
239 0331 2 (MASK NUMERIC),
240 0332 2 REP CHAR ONE - CHAR_ZERO - 1 OF (0),
241 0333 2 (MASK NUMERIC),
242 0334 2 REP CHAR TWO - CHAR_ONE - 1 OF (0),
243 0335 2 (MASK NUMERIC),
244 0336 2 REP CHAR THREE - CHAR_TWO - 1 OF (0),
245 0337 2 (MASK NUMERIC),
246 0338 2 REP CHAR FOUR - CHAR_THREE - 1 OF (0),
247 0339 2 (MASK NUMERIC),
248 0340 2 REP CHAR FIVE - CHAR_FOUR - 1 OF (0),
249 0341 2 (MASK NUMERIC),
250 0342 2 REP CHAR SIX - CHAR_FIVE - 1 OF (0),
251 0343 2 (MASK NUMERIC),
252 0344 2 REP CHAR SEVEN - CHAR_SIX - 1 OF (0),
253 0345 2 (MASK NUMERIC),
254 0346 2 REP CHAR EIGHT - CHAR_SEVEN - 1 OF (0),
255 0347 2 (MASK NUMERIC),
256 0348 2 REP CHAR NINE - CHAR_EIGHT - 1 OF (0),
257 0349 2 (MASK NUMERIC),
258 0350 2 REP 255 - CHAR NINE OF (0)
259 0351 2 ) : VECTOR [256, BYTE, UNSIGNED];
260 0352
261 0353 LOCAL
262 0354 USER_FLAGS,
263 0355 TEMP_LENGTH : INITIAL (1),
264 0356
265 0357 TEXT_BUF : VECTOR [32, BYTE],
266 0358 INT_DIGITS,
267 0359
268 0360 TOTAL_INT_DIGITS : INITIAL (0),
269 0361 ACTUAL_SCALE : INITIAL (0),
270 0362
271 0363 TOTAL_FRAC_DIGITS : INITIAL (0),
272 0364 DIFF,
273 0365 CURRENT_PTR,
274 0366 ROUND_DIGIT : INITIAL (0),
275 0367 MASK,
276 0368 START_PTR,
277 0369 DEC_PT_LOC : INITIAL (0),
278 0370 END_TXT_PTR;
279 0371
280 0372 MAP
281 0373 CURRENT_PTR : REF VECTOR [,BYTE];
282 0374
283 0375
284 0376 !+ The overall strategy is to construct an acceptable numeric string for CVTSP
285 0377 in a temporary area, TEXT_BUF. "Acceptable" means no embedded spaces, tabs,
286 0378 or decimal point, and less than or equal to 31 digits. This routine should
287 0379 detect any invalid strings which would cause a reserved operand from CVTSP.
288 0380 !-
289 0381
290 0382 IF ACTUALCOUNT () LSS K_FLAGS_ARG
291 0383 THEN
292 0384 USER_FLAGS = 0
293 0385 ELSE
294 0386 USER_FLAGS = .FLAGS;
295 0387 2 ! store caller's flags, if any
```

```
296 0388 2 END_TXT_PTR = .TEXT_DSC [DSC$A_POINTER] + .TEXT_DSC [DSC$W_LENGTH];
297 0389
298 0390
299 0391 + Skip leading spaces. Skip leading tabs only if bit in flag is set.
300 0392 -
301 0393
302 0394 IF (.USER_FLAGS AND V_SKIP_TABS) NEQ 0
303 0395 THEN
304 0396 MASK = MASK_BLANKS OR MASK_TAB
305 0397 ELSE
306 0398 MASK = MASK_BLANKS;
307 0399 CURRENT_PTR = SPANC (TEXT_DSC [DSC$W_LENGTH], .TEXT_DSC [DSC$A_POINTER],
308 0400 SPANC_TABLE, MASK);
309 0401
310 0402 +
311 0403 Always insert a sign into the temporary area so that we know a sign
312 0404 is always included in TEMP_LENGTH.
313 0405 -
314 0406
315 0407 IF .CURRENT_PTR NEQ 0 AND
316 0408 (.CURRENT_PTR [0] EQL %C'+' OR
317 0409 .CURRENT_PTR [0] EQL %C'-' )
318 0410 THEN
319 0411 BEGIN
320 0412 TEXT_BUF [0] = .CURRENT_PTR [0];
321 0413 CURRENT_PTR = .CURRENT_PTR + 1;
322 0414 END
323 0415 ELSE
324 0416 BEGIN
325 0417 TEXT_BUF [0] = %C'+';
326 0418 END;
327 0419
328 0420 START_PTR = .CURRENT_PTR; ! start = 1st possible digit
329 0421
330 0422
331 0423 +
332 0424 Search for integer digits. Embedded tabs and spaces may be ignored, depending
333 0425 on USER_FLAGS.
334 0426 -
335 0427
336 0428 WHILE 1 DO
337 0429 BEGIN
338 0430
339 0431 +
340 0432 Wait until here to check for all blanks in the input string.
341 0433 A zero string must contain the proper number of fractional digits,
342 0434 so it's easiest to let zeroes be handled the same as non-zeroes.
343 0435 -
344 0436
345 0437 IF .CURRENT_PTR EQL 0
346 0438 THEN
347 0439 BEGIN
348 0440 TOTAL_INT_DIGITS = 1;
349 0441 TEXT_BUF [1] = %C'0';
350 0442 TEMP_LENGTH = .TEMP_LENGTH + 1;
351 0443 EXIT[LOOP]
352 0444 END;
```

```
353 0445 3
354 0446 3
355 0447 3
356 0448 3
357 0449 3
358 0450 3
359 0451 3
360 0452 3
361 0453 3
362 0454 3
363 0455 3
364 0456 3
365 0457 3
366 0458 3
367 0459 3
368 0460 3
369 0461 3
370 0462 3
371 0463 3
372 0464 3
373 0465 3
374 0466 3
375 0467 3
376 0468 3
377 0469 3
378 0470 3
379 0471 3
380 0472 3
381 0473 3
382 0474 3
383 0475 3
384 0476 3
385 0477 3
386 0478 3
387 0479 3
388 0480 3
389 0481 3
390 0482 3
391 0483 3
392 0484 3
393 0485 3
394 0486 3
395 0487 3
396 0488 3
397 0489 3
398 0490 3
399 0491 3
400 0492 3
401 0493 3
402 0494 3
403 0495 3
404 0496 3
405 0497 3
406 0498 3
407 0499 3
408 0500 3
409 0501 3

CURRENT_PTR = SPANC (XREF (.END_TXT_PTR - .START_PTR),
                     .CURRENT_PTR, SPANC_TABLE, XREF (MASK_NUMERIC));

!+ Store any digits found.
!-

INT_DIGITS = (IF .CURRENT_PTR NEQ 0
              THEN .CURRENT_PTR - .START_PTR
              ELSE .END_TXT_PTR - .START_PTR);
! calc # integer digits

IF .INT_DIGITS GTR 0
THEN
  BEGIN
    CHSMOVE (.INT_DIGITS, .START_PTR, TEXT_BUF + .TEMP_LENGTH);
    ! copy int. digits to temp area
    TEMP_LENGTH = .TEMP_LENGTH + .INT_DIGITS;
    TOTAL_INT_DIGITS = .TOTAL_INT_DIGITS + .INT_DIGITS;
  END;

IF .CURRENT_PTR NEQ 0
THEN
  BEGIN
    !+ Have found a non-digit. It could be a decimal point or embedded blanks, etc.
    !-
    SELECTONE .CURRENT_PTR [0] OF
      SET
        [XC'.']:
          BEGIN
            DEC_PT_LOC = .CURRENT_PTR; ! save location of dec pt
            CURRENT_PTR = .CURRENT_PTR + 1;
          EXITLOOP
          END;
        [CHAR SPACE, CHAR_NULL]:
          BEGIN
            IF (.USER_FLAGS AND V_SKIP_BLANKS) NEQ 0
            THEN
              BEGIN
                CURRENT_PTR = SPANC (XREF (.END_TXT_PTR - .START_PTR),
                                    .CURRENT_PTR, SPANC_TABLE,
                                    XREF (MASK_BLANKS));
                IF .CURRENT_PTR EQL 0 THEN EXITLOOP;
                START_PTR = .CURRENT_PTR;
              END
            ELSE
              RETURN 0;
            ! <should be zeroes here>
          END;
        [CHAR TAB]:
          BEGIN
            IF (.USER_FLAGS AND V_SKIP_TABS) NEQ 0
            THEN
              BEGIN
                CURRENT_PTR = SPANC (XREF (.END_TXT_PTR - .START_PTR),
                                    .CURRENT_PTR, SPANC_TABLE,
                                    XREF (MASK_TAB));
```



BAS\$CVT\_T\_P  
1-010

N 10  
BAS\$CVT\_T\_P - convert numeric text to packed de 16-Sep-1984 00:17:11  
BAS\$CVT\_T\_P - convert numeric text to packed de 14-Sep-1984 11:54:49

VAX-11 Bliss-32 V4.0-742  
[BASRTL.SRC]BAS\$CVTTP.B32;1

Page 10  
(3)

```
410 0502 6 IF .CURRENT_PTR EQL 0 THEN EXITLOOP;
411 0503 6 START_PTR = .CURRENT_PTR;
412 0504 6 END
413 0505 3 ELSE
414 0506 3 RETURN 0;
415 0507 4 END;
416 0508 4 [OTHERWISE]:
417 0509 4 RETURN 0;
418 0510 4 ! no other chars valid here
419 0511 4 TES;
420 0512 4 END
421 0513 4 ELSE
422 0514 4 !+
423 0515 4 Only digits found. Purely integer number.
424 0516 4 -
425 0517 4 EXITLOOP;
426 0518 4
427 0519 4 END;
428 0520 4 ! end of WHILE loop
429 0521 4 !+
430 0522 4 Return an error if the number of integer digits exceeds what the destination
431 0523 4 descriptor specifies.
432 0524 4 -
433 0525 4
434 0526 4 IF .TOTAL_INT_DIGITS GTR (.PACKED_DSC [DSC$W_LENGTH] +
435 0527 4 .PACKED_DSC [DSC$B_SCALE])
436 0528 4 THEN RETURN 0;
437 0529 4
438 0530 4 !+
439 0531 4 Search for fractional digits if a decimal point was found and we are not
440 0532 4 at the end of the string.
441 0533 4 -
442 0534 4
443 0535 4 IF .DEC_PT_LOC NEQ 0 AND
444 0536 4 .CURRENT_PTR NEQ 0 AND
445 0537 4 .CURRENT_PTR LSS .END_TXT_PTR
446 0538 4 THEN
447 0539 4 BEGIN
448 0540 4
449 0541 4 START_PTR = .DEC_PT_LOC + 1;
450 0542 4
451 0543 4 WHILE 1 DO
452 0544 4 BEGIN
453 0545 4 CURRENT_PTR = SPANC (%REF (.END_TXT_PTR - .START_PTR),
454 0546 4 .CURRENT_PTR, SPANC_TABLE, %REF (MASK_NUMERIC));
455 0547 4
456 0548 4 !+
457 0549 4 Copy any digits found into temp. area.
458 0550 4 -
459 0551 4
460 0552 4 ACTUAL_SCALE = (IF .CURRENT_PTR NEQ 0
461 0553 4 THEN .CURRENT_PTR - .START_PTR
462 0554 4 ELSE .END_TXT_PTR - .START_PTR);
463 0555 4 ! calc # of fract digits
464 0556 4 IF .ACTUAL_SCALE NEQ 0
465 0557 4 THEN
466 0558 4 BEGIN
```

```
467 0559 5 CHSMOVE (.ACTUAL_SCALE, .START_PTR, TEXT_BUF + .TEMP_LENGTH);
468 0560 5 ! copy fract to temp area
469 0561 5 TEMP_LENGTH = .TEMP_LENGTH + .ACTUAL_SCALE;
470 0562 5 TOTAL_FRAC_DIGITS = .TOTAL_FRAC_DIGITS + .ACTUAL_SCALE;
471 0563 4 END;
472 0564 4
473 0565 4 IF .CURRENT_PTR NEQ 0 AND
474 0566 4 .CURRENT_PTR LSS .END_TXT_PTR
475 0567 4 ! at end of string?
476 0568 4 THEN
477 0569 5 BEGIN
478 0570 5 +
479 0571 5 Have found a non-digit. It may be an embedded blank or tab, or trailing
480 0572 5 blank or tab.
481 0573 5 -
482 0574 5 SELECTONE .CURRENT_PTR [0] OF
483 0575 5 SET
484 0576 5 [CHAR SPACE, CHAR_NULL]:
485 0577 6 BEGIN
486 0578 6 IF (.USER_FLAGS AND V_SKIP_BLANKS) NEQ 0
487 0579 6 THEN
488 0580 7 BEGIN
489 0581 7 CURRENT_PTR = SPANC (%REF (.END_TXT_PTR - .START_PTR),
490 0582 7 .CURRENT_PTR, SPANC_TABLE,
491 0583 7 %REF (MASK_BLANKS));
492 0584 7 IF .CURRENT_PTR EQL 0 THEN EXITLOOP;
493 0585 7 START_PTR = .CURRENT_PTR;
494 0586 7 END
495 0587 6 ELSE
496 0588 6 RETURN 0; ! ret an error
497 0589 5 END;
498 0590 5 [CHAR TAB]:
499 0591 6 BEGIN
500 0592 6 IF (.USER_FLAGS AND V_SKIP_TABS) NEQ 0
501 0593 6 THEN
502 0594 7 BEGIN
503 0595 7 CURRENT_PTR = SPANC (%REF (.END_TXT_PTR - .START_PTR),
504 0596 7 .CURRENT_PTR, SPANC_TABLE,
505 0597 7 %REF (MASK_TAB));
506 0598 7 IF .CURRENT_PTR EQL 0 THEN EXITLOOP;
507 0599 7 START_PTR = .CURRENT_PTR;
508 0600 7 END
509 0601 6 ELSE
510 0602 6 RETURN 0;
511 0603 5 END;
512 0604 5 [OTHERWISE]:
513 0605 5 RETURN 0; ! invalid character
514 0606 5 TES;
515 0607 5
516 0608 5 END
517 0609 5
518 0610 4 ELSE
519 0611 4 +
520 0612 4 Fraction digits were the rest of the string.
521 0613 4 -
522 0614 4 EXITLOOP;
523 0615 4
```

```
524 0616      END;                                ! End of 'WHILE' loop
525 0617
526 0618      END;                                ! end of fract. digit search
527 0619
528 0620      + Does the input text use all the fractional digits allowed by the decimal
529 0621      descriptor?
530 0622      -
531 0623
532 0624      DIFF = (-.PACKED_DSC [DSC$B_SCALE]) - .TOTAL_FRAC_DIGITS;
533 0625      IF .DIFF GTR 0
534 0626      THEN
535 0627          BEGIN
536 0628      +
537 0629      - Fewer fractional digits than allowed. Pad with zeroes.
538 0630
539 0631          CH$COPY (.TEMP_LENGTH, TEXT_BUF, %C'0', .TEMP_LENGTH + .DIFF,
540 0632          TEXT_BUF);
541 0633          TEMP_LENGTH = .TEMP_LENGTH + .DIFF;
542 0634      END
543 0635
544 0636      ELSE IF .DIFF LSS 0
545 0637      THEN
546 0638          BEGIN
547 0639      +
548 0640      - Too many fractional digits. Save the first digit past the allowed number
549 0641      of fractional digits for rounding. The value will be truncated if the user
550 0642      set dont round in USER_FLAGS, or if the digit to round on is less than five.
551 0643      Note that if the caller passed an input string longer than 31 digits, the
552 0644      32nd digit is saved for rounding purposes. (It is lost after the CVISP
553 0645      instruction.)
554 0646
555 0647      -
556 0648
557 0649          LOCAL
558 0650          INDEX;
559 0651
560 0652          INDEX = 1                                ! for sign
561 0653          + .TOTAL_INT_DIGITS
562 0654          + .TOTAL_FRAC_DIGITS
563 0655          - 1;                                     ! start counting from 0
564 0656          ROUND_DIGIT = .TEXT_BUF [INDEX];        ! save digit to round on
565 0657          TEMP_LENGTH = .TEMP_LENGTH + .DIFF;      ! subtr extra fract digits
566 0658                                                  ! from text string
567 0659
568 0660      END;
569 0661
570 0662      + Copy the text to the destination descriptor and convert to packed at the
571 0663      same time.
572 0664      -
573 0665
574 0666
575 0667      TEMP_LENGTH = MIN (.TEMP_LENGTH - 1, K_MAX_DIGITS);
576 0668                                                  ! not more than 31 digits and
577 0669                                                  ! don't count sign
578 0670      CVTSP (TEMP_LENGTH, TEXT_BUF, PACKED_DSC [DSC$W_LENGTH],
579 0671      .PACKED_DSC [DSC$A_POINTER]);
580 0672
```



```
581 0673 2
582 0674 2
583 0675 2
584 0676 2
585 0677 2
586 0678 2
587 0679 2
588 0680 2
589 0681 2
590 0682 2
591 0683 2
592 0684 2
593 0685 2
594 0686 2
595 0687 2
596 0688 2
597 0689 2
598 0690 2
599 0691 2
600 0692 2
601 0693 2
602 0694 2
603 0695 2
604 0696 2
605 0697 2
606 0698 2
607 0699 1

+ see if we need to round this number. the number will be rounded if the
+ user did a SET ROUND and there is a rounding digit >= 5.
-
IF (.USER_FLAGS AND V_DONT_ROUND) EQL 0 AND
  .ROUND_DIGIT GEQ %C'5'
THEN
  BEGIN
  LOCAL
  ONE:
  IF .TEXT_BUF [0] EQL %C'-'
  THEN
    ONE = K_PACKED_NEG_ONE
  ELSE
    ONE = K_PACKED_ONE;
+
+ add one to the last significant digit in the number to cause
+ the rounding effect.
-
IF NOT ADDP (1, ONE, .PACKED_DSC [DSC$W_LENGTH], .PACKED_DSC [DSC$A_POINTER])
THEN RETURN 0;
END;

RETURN 1;

END;
```

! End of routine BASSCVT\_T\_P

.TITLE BASSCVT\_T\_P BASSCVT\_T\_P - convert numeric text  
to packed de

.IDENT \1-010\

.PSECT \_BASSCODE, NOWRT, SHR, PIC, 2

```
02 00000 P.AAA: .BYTE 2
00# 00001 .BYTE 0[8]
04 00009 .BYTE 4
00# 0000A .BYTE 0[22]
02 00020 .BYTE 2
00# 00021 .BYTE 0[15]
01 00030 .BYTE 1
01 00031 .BYTE 1
01 00032 .BYTE 1
01 00033 .BYTE 1
01 00034 .BYTE 1
01 00035 .BYTE 1
01 00036 .BYTE 1
01 00037 .BYTE 1
01 00038 .BYTE 1
01 00039 .BYTE 1
00# 0003A .BYTE 0[198]
```

SPANC\_TABLE= P.AAA  
.EXTRN LIB\$\$ADDP\_R7

OFFC 00000

.ENTRY BASSCVT\_T\_P, Save R2,R3,R4,R5,R6,R7,R8,R9,- : 0218

			5E	BC	AE	9E	00002		MOVAB	R10, R11		
			56		01	D0	00006		MOVL	-68(SP), SP		
				10	AE	7C	00009		CLRQ	#1, TEMP_LENGTH	0284	
				08	AE	7C	0000C		CLRQ	ACTUAL_SCALE		
				04	AE	D4	0000F		CLRL	ROUND_DIGIT		
			03		6C	91	00012		CMPB	DEC_PT_LOC		
					04	1E	00015		BRB	(APT, #3	0382	
					6E	D4	00017		BGEQU	1\$		
					04	11	00019		CLRL	USER_FLAGS	0384	
			6E	OC	AC	D0	0001B	1\$:	BRB	2\$		
			50	04	AC	D0	0001F	2\$:	MOVL	FLAGS, USER_FLAGS	0386	
			57		60	3C	00023		MOVL	TEXT_DSC, R0	0388	
			57	04	A0	C0	00026		MOVZWL	(R0), END_TXT_PTR		
				1C	AE	D4	0002A		ADDL2	4(R0), END_TXT_PTR		
	08		6E		04	E1	0002D		CLRL	28(SP)	0394	
				1C	AE	D6	00031		BBC	#4, USER_FLAGS, 3\$		
			52		06	D0	00034		INCL	28(SP)		
					03	11	00037		MOVL	#6, MASK	0396	
			52		02	D0	00039	3\$:	BRB	4\$		
52	FE8D	CF	04	B0	60	2B	0003C	4\$:	MOVL	#2, MASK	0398	
					02	12	00044		SPANC	(R0), 24(R0), SPANC_TABLE, MASK	0399	
					51	D4	00046		BNEQ	5\$		
			59		51	D0	00048	5\$:	CLRL	R1		
					10	13	0004B		MOVL	R1, CURRENT_PTR		
			2B		69	91	0004D		BEQL	7\$	0407	
					05	13	00050		CMPB	(CURRENT_PTR), #43	0408	
			2D		69	91	00052		BEQL	6\$		
					06	12	00055		CMPB	(CURRENT_PTR), #45	0409	
			24	AE	89	90	00057	6\$:	BNEQ	7\$		
					04	11	0005B		MOVB	(CURRENT_PTR)+, TEXT_BUF	0412	
			24	AE	2B	90	0005D	7\$:	BRB	8\$	0407	
			5B		59	D0	00061	8\$:	MOVB	#43, TEXT_BUF	0417	
					0C	12	00064		MOVL	CURRENT_PTR, START_PTR	0420	
			14	AE	01	D0	00066		BNEQ	9\$	0437	
			25	AE	30	90	0006A		MOVL	#1, TOTAL_INT_DIGITS	0440	
					56	D6	0006E		MOVB	#48, TEXT_BUF+1	0441	
					79	11	00070		INCL	TEMP_LENGTH	0442	
			57		5B	C3	00072	9\$:	BRB	21\$	0439	
01	FE84	5A CF	69		5A	2B	00076		SUBL3	START_PTR, END_TXT_PTR, R10	0446	
					02	12	0007D		SPANC	R10, (CURRENT_PTR), SPANC_TABLE, #1	0447	
					51	D4	0007F		BNEQ	10\$		
			59		51	D0	00081	10\$:	CLRL	R1		
				18	AE	D4	00084		MOVL	R1, CURRENT_PTR	0446	
					59	D5	00087		CLRL	24(SP)	0453	
					09	13	00089		TSTL	CURRENT_PTR		
				18	AE	D6	0008B		BEQL	11\$		
			59		5B	C3	0008E		INCL	24(SP)		
	58				03	11	00092		SUBL3	START_PTR, CURRENT_PTR, INT_DIGITS	0454	
			58		5A	D0	00094	11\$:	BRB	12\$		
					0D	15	00097	12\$:	MOVL	R10, INT_DIGITS	0455	
	24 AE46		6B		5B	28	00099		BLEQ	13\$	0457	
					58	C0	0009F		MOV3	INT_DIGITS, (START_PTR), TEXT_BUF-	0460	
			14	56	58	C0	000A2		ADDL2	[TEMP_LENGTH]		
			41	AE	58	C0	000A6	13\$:	ADDL2	INT_DIGITS, TEMP_LENGTH	0462	
			2E		AE	E9	000AA		ADDL2	INT_DIGITS, TOTAL_INT_DIGITS	0463	
					69	91	000AA		BLBC	24(SP), 21\$	0466	
									CMPB	(CURRENT_PTR), #46	0474	

BASSCVT\_T\_P  
1-010

BASSCVT\_T\_P - convert numeric text to packed de  
BASSCVT\_T\_P - convert numeric text to packed de

F 11

16-Sep-1984 00:17:11  
14-Sep-1984 11:54:49

VAX-11 Bliss-32 V4.0-742  
[BASRTL.SRC]BASSCVTTP.B32;1

Page 15  
(3)

04	AE	06	12	000AD	BNEQ	14\$				
		89	9E	000AF	MOVAB	(CURRENT_PTR)+, DEC_PT_LOC		0476		
		36	11	000B3	BRB	21\$		0475		
		69	95	000B5	14\$: TSTB	(CURRENT_PTR)		0480		
		05	13	000B7	BEQL	15\$				
	20	69	91	000B9	CMPB	(CURRENT_PTR), #32				
		11	12	000BC	BNEQ	18\$				
	03	6E	E8	000BE	15\$: BLBS	USER_FLAGS, 17\$		0482		
		0142	31	000C1	16\$: BRW	43\$				
02	FE36	CF	69	5A	2B	000C4	17\$: SPANC	R10, (CURRENT_PTR), SPANC_TABLE, #2	0487	
			14	13	000CB	BEQL	19\$			
			14	11	000CD	BRB	20\$	0485		
	09	69	91	000CF	18\$: CMPB	(CURRENT_PTR), #9		0494		
		ED	12	000D2	BNEQ	16\$				
04	FE22	CF	E9	1C	AE	E9	000D4	28(SP), 16\$	0496	
			69	5A	2B	000D8	SPANC	R10, (CURRENT_PTR), SPANC_TABLE, #4	0501	
				02	12	000DF	BNEQ	20\$		
			59	51	D4	000E1	19\$: CLRL	R1		
				51	D0	000E3	20\$: MOVL	R1, CURRENT_PTR	0499	
				03	13	000E6	BEQL	21\$	0502	
			58	FF	76	31	000E8	8\$		
			50	08	AC	D0	000EB	21\$: MOVL	PACKED_DSC, R8	0526
			51	08	68	3C	000EF	MOVZWL	(R8), R0	0527
			50		AB	98	000F2	CVTBL	8(R8), R1	
			50		51	C0	000F6	ADDL2	R1, R0	
				14	AE	D1	000F9	CMPL	TOTAL_INT_DIGITS, R0	0526
					C2	14	000FD	BGTR	16\$	
			04	AE	D5	000FF	TSTL	DEC_PT_LOC		0535
				02	13	00102	BEQL	22\$		
			59	D5	00104	TSTL	CURRENT_PTR			0536
			57	7D	13	00106	22\$: BEQL	34\$		
				59	D1	00108	CMPL	CURRENT_PTR, END_TXT_PTR		0537
				7F	18	0010B	BGEQ	35\$		
		5B	04	AE	01	C1	0010D	ADDL3	#1, DEC_PT_LOC, START_PTR	0541
		SA	57	5B	C3	00112	23\$: SUBL3	START_PTR, END_TXT_PTR, R10		0545
01	FDE4	CF	69	5A	2B	00116	SPANC	R10, (CURRENT_PTR), SPANC_TABLE, #1		0546
				02	12	0011D	BNEQ	24\$		
			59	51	D4	0011F	CLRL	R1		
				51	D0	00121	24\$: MOVL	R1, CURRENT_PTR		0545
				18	AE	D4	00124	CLRL	24(SP)	0552
				59	D5	00127	TSTL	CURRENT_PTR		
				0A	13	00129	BEQL	25\$		
			59	18	AE	D6	0012B	INCL	24(SP)	
	10	AE		5B	C3	0012E	SUBL3	START_PTR, CURRENT_PTR, ACTUAL_SCALE		0553
				04	11	00133	BRB	26\$		
			10	AE	5A	D0	00135	25\$: MOVL	R10, ACTUAL_SCALE	0554
				10	13	00139	26\$: BEQL	27\$		0556
	24	AE46	6B	10	AE	28	0013B	MOV3	ACTUAL_SCALE, (START_PTR), TEXT_BUF-[TEMP_LENGTH]	0559
			56	10	AE	C0	00142	ADDL2	ACTUAL_SCALE, TEMP_LENGTH	0561
			OC	AE	C0	00146	ADDL2	ACTUAL_SCALE, TOTAL_FRAC_DIGITS		0562
			3D	18	AE	E9	0014B	27\$: BLBC	24(SP), 35\$	0565
			57	59	D1	0014F	CMPL	CURRENT_PTR, END_TXT_PTR		0566
				38	18	00152	BGEQ	35\$		
				69	95	00154	TSTB	(CURRENT_PTR)		0576
				05	13	00156	BEQL	28\$		
			20	69	91	00158	CMPB	(CURRENT_PTR), #32		



BASSCVT\_T\_P  
1-010

BASSCVT\_T\_P - convert numeric text to packed de 16-Sep-1984 00:17:11  
BASSCVT\_T\_P - convert numeric text to packed de 14-Sep-1984 11:54:49

VAX-11 Bliss-32 V4.0-742  
[BASRTL.SRC]BASCVTTP.B32;1

Page 16  
(3)

				11	12	00158		BNEQ	31\$			
		03		6E	E8	0015D	28\$:	BLBS	USER_FLAGS, 30\$			0578
				00A3	31	00160	29\$:	BRW	43\$			
02	FD97	CF	69	5A	2B	00163	30\$:	SPANC	R10, (CURRENT_PTR), SPANC_TABLE, #2			0583
				14	13	0016A		BEQL	32\$			
				14	11	0016C		BRB	33\$			0581
		09		69	91	0016E	31\$:	CMPB	(CURRENT_PTR), #9			0590
				ED	12	00171		BNEQ	29\$			
04	FD83	CF	E9	1C	AE	E9	00173	BLBC	28(SP), 29\$			0592
			69		5A	2B	00177	SPANC	R10, (CURRENT_PTR), SPANC_TABLE, #4			0597
					02	12	0017E	BNEQ	33\$			
					51	D4	00180	CLRL	R1			
			59		51	D0	00182	MOVL	R1, CURRENT_PTR			0595
					05	13	00185	34\$:	BEQL	35\$		0598
			5B		59	D0	00187	MOVL	CURRENT_PTR, START_PTR			0599
					86	11	0018A	BRB	23\$			0592
			57	08	A8	98	0018C	35\$:	CVTBL	8(R8), R7		0624
			57	0C	AE	C0	00190	ADDL2	TOTAL_FRAC_DIGITS, R7			
			57		57	CE	00194	MNEGL	R7, DIFF			
					0E	15	00197	BLEQ	36\$			0625
					57	C1	00199	ADDL3	DIFF, TEMP_LENGTH, R0			0632
50		50	24	AE	56	2C	0019D	MOVCS	TEMP_LENGTH, TEXT_BUF, #48, R0, TEXT_BUF			
		30			AE		001A3					
					11	11	001A5	BRB	37\$			0634
					12	18	001A7	36\$:	BGEQ	38\$		0637
			51	14	AE	D0	001A9	MOVL	TOTAL_INT_DIGITS, R1			0655
		50	51	0C	AE	C1	001AD	ADDL3	TOTAL_FRAC_DIGITS, R1, INDEX			
			08	AE	24	AE	40	9A	001B2	MOVZBL	TEXT_BUF[INDEX], ROUND_DIGIT	0656
			56		57	C0	001B8	37\$:	ADDL2	DIFF, TEMP_LENGTH		0658
			50	FF	A6	9E	001BB	38\$:	MOVAB	-1(R6), R0		0668
			1F		50	D1	001BF	CMPL	R0, #31			
					03	15	001C2	BLEQ	39\$			
			50		1F	D0	001C4	MOVL	#31, R0			
			56		50	D0	001C7	39\$:	MOVL	R0, TEMP_LENGTH		
04	B8	68	24	AE	56	09	001CA	CVTSP	TEMP_LENGTH, TEXT_BUF, (R8), #4(R8)			0672
		2D			03	E0	001D1	BBS	#3, USER_FLAGS, 42\$			0678
			35	08	AE	D1	001D5	CMPL	ROUND_DIGIT, #53			0679
					27	19	001D9	BLSS	42\$			
			2D	24	AE	91	001DB	CMPB	TEXT_BUF, #45			0684
					06	12	001DF	BNEQ	40\$			
			20	AE	1D	D0	001E1	MOVL	#29, ONE			0686
					04	11	001E5	BRB	41\$			
			20	AE	1C	D0	001E7	40\$:	MOVL	#28, ONE		0688
					55	20	AE	9E	001EB	41\$:		0693
					57	04	A8	D0	001EF			
					56	68	3C	001F3	MOVZWL	(R8), R6		
					54	01	D0	001F6	MOVL	#1, R4		
					00	16	001F9	JSB	LIB\$\$ADDP_R7			
			04		50	E9	001FF	BLBC	R0, 43\$			
			50		01	D0	00202	42\$:	MOVL	#1, R0		0697
						04	00205	RET				
					50	D4	00206	43\$:	CLRL	R0		0699
						04	00208	RET				

; Routine Size: 521 bytes, Routine Base: \_BAS\$CODE + 0100

BASSCVT\_T\_P  
 1-010

H 11  
 BASSCVT\_T\_P - convert numeric text to packed de 16-Sep-1984 00:17:11  
 BASSCVT\_T\_P - convert numeric text to packed de 14-Sep-1984 11:54:49

VAX-11 Bliss-32 V4.0-742  
 [BASRTL.SRC]BASCVTTP.B32;1

Page 17  
 (3)

```

: 608      0700 1
: 609      0701 1 END
: 610      0702 1
: 611      0703 0 ELUDOM
    
```

! End of module BASSCVT\_T\_P

PSECT SUMMARY

Name	Bytes	Attributes
_BASSCODE	777	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_S255SDUA28:[SYSLIB]STARLET.L32;1	9776	3	0	581	00:01.1

COMMAND QUALIFIERS

```

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LISS:BASCVTTP/OBJ=OBJ$:BASCVTTP MSRC$:BASCVTTP/UPDATE=(ENH$:BASCVTTP)
:
: Size:          521 code + 256 data bytes
: Run Time:      00:15.2
: Elapsed Time:  00:35.7
: Lines/CPU Min: 2780
: Lexemes/CPU-Min: 12506
: Memory Used:   203 pages
: Compilation Complete
    
```



0021 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY